build.gradle[rediger]

```
dependencies {
  implementation 'org.springframework.session:spring-session-core'
  implementation "org.springframework.session:spring-session-data-redis"
  implementation 'redis.clients:jedis'
}
```

application.yml[rediger]

```
spring:
        session:
                store-type: redis
#optionally define namespace for your application
                redis:
                        namespace: yourapp
        redis:
                host: your-redis-server-dns-name
                password: your-redis-password
                port: 6379
```

Flash and redis alternative 1[rediger]

src/main/java/no.prpr.springsession.HttpSessionSynchronizer[rediger]

```
package no.prpr.springsession;

import org.springframework.core.annotation.Order;
```

```java
import org.springframework.session.web.http.SessionRepositoryFilter;
import org.springframework.util.Assert;
import org.springframework.web.filter.OncePerRequestFilter;

import javax.servlet.FilterChain;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;
import java.util.Enumeration;

/**
 * @author jitendra
 */
@Order(SessionRepositoryFilter.DEFAULT_ORDER + 1)
public class HttpSessionSynchronizer extends OncePerRequestFilter {

    private Boolean persistMutable;


    @Override
    public void afterPropertiesSet() throws ServletException {
        super.afterPropertiesSet();
        Assert.notNull(persistMutable);
    }

    @Override
    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain
filterChain) throws ServletException, IOException {
```

```
        filterChain.doFilter(request, response);
        if (persistMutable && request != null && request.getSession(false) != null) {
            HttpSession session = request.getSession();
            Enumeration<String> attributeNames = session.getAttributeNames();
            while (attributeNames.hasMoreElements()) {
                String key = attributeNames.nextElement();
                try {
                    Object object = session.getAttribute(key);
                    session.setAttribute(key, object);
                } catch (Exception ignored) {
                }
            }
        }
    }

    public void setPersistMutable(Boolean persistMutable) {
        this.persistMutable = persistMutable;
    }
}
```

resources.groovy[rediger]

```
Beans = {

    // append at end
    httpSessionSynchronizer(HttpSessionSynchronizer) {
        persistMutable = true // conf.allow.persist.mutable as Boolean
    }
}
```