

Classroom Booking Made Easy

FOUNDERS



NIHESH ANDERSON, sophomore, IIITD



HARSH PATHAK, sophomore, IIITD

Design Patterns



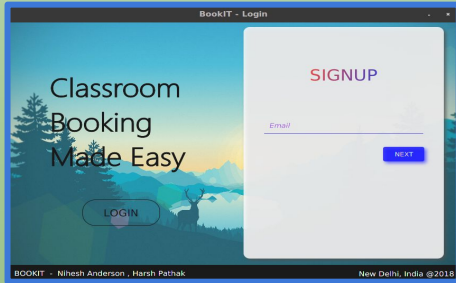
Observer Design Pattern

- Faculty and student - Enroll into courses
- Latest course state is accessed by the User Objects
- Changes to course - Immediately reflected on Faculty and User

```
nireshgandy-yoga710 ~/Documents/Andy's Linux/BookIT/BookIT_server $ ./BookIT.sh
Setting up spam filter...
Spam filter has been set up.
[ 2017-11-16T10:55:38.179 ] /192.168.59.171 | Performing GetUser
[ 2017-11-16T10:55:39.915 ] /192.168.59.171 | Performing GetUser
[ 2017-11-16T10:55:45.743 ] /192.168.59.171 | Performing AllCourses
[ 2017-11-16T10:55:49.287 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.459 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.579 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.651 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.715 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.779 ] /192.168.59.171 | Performing ReadCourse
[ 2017-11-16T10:55:49.811 ] /192.168.59.171 | Performing WriteUser
[ 2017-11-16T10:56:43.827 ] /192.168.59.171 | Performing ReadRoom
[ 2017-11-16T10:56:43.971 ] /192.168.59.171 | Performing ReadRoom
```

State Design Pattern

- Server works differently in different states
- The client sets server state and fetches required information
- Ex: Read Course, Get User, Write User



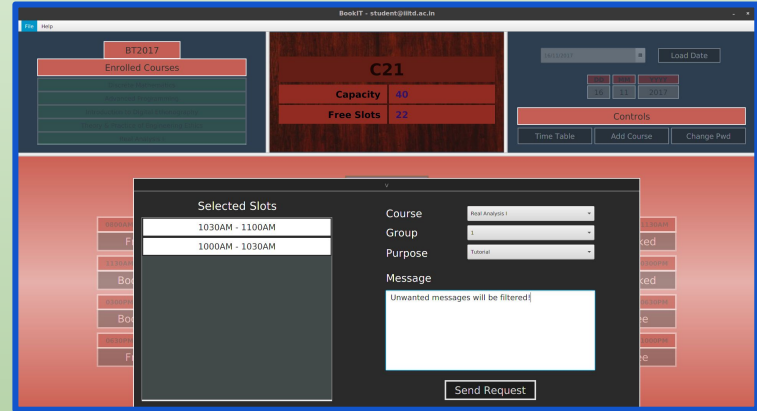
Chain of Responsibility Design Pattern

- Login/Signup - User goes through multiple validation stages
- Each stage handles different aspects of Login/Signup
- Ex: Entering email -> Entering password -> Login
- Ex: Entering email -> pass and confirm pass -> Join Code, etc.

Other design patterns include Iterator and Singleton

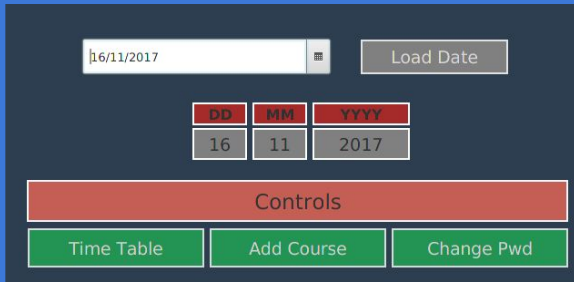
Challenges

- Getting the app to work on **various platforms** and on **different screen resolutions**.
- Implementing **networking based app** so that **multiple devices** can access the app from a **common server** without conflicts.
- Implementing spam filter on the app to identify spam messages during send requests with a high **TPR (true positive rate)**
- Providing a generic model of the app which can be easily extended at any time to support time table of other colleges **if the project is scaled**.
- Coming up with a innovative GUI design and a proper logic of the working of the app in the given time.
- Resolving minor bugs/glitches that could potentially have been misused to tamper the database



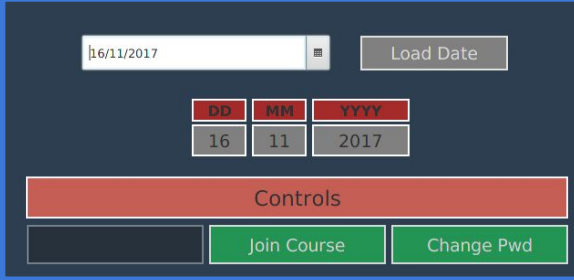
Major Highlights

1. Networking based architecture - users can access the app from their devices anywhere in IIIT Delhi or using a VPN
2. Spam Filter - The app detects spam message requests and deletes them automatically, keeping the admins at ease.
3. Packaging - The project has been bundled into client JAR and server JAR for one click launch of the application. Documentation attached
4. Cross platform compatibility - The application works on both windows and linux platforms irrespective of the screen resolution of the target PC.
5. Innovative Design and GUI - The GUI of the app is different from what you normally see in a drag and drop made app in Scenebuilder. Easy to use interface
6. Change Password - Users can maximise the security by changing their passwords.
7. Joining Code facility - The additional join code generating facility allows admin to generate codes which then can be shared by the admin to prospective users. This prevents the app from being misused.
8. Run the app on windows using an executable file - The executable file not only makes it easy but also looks professional.



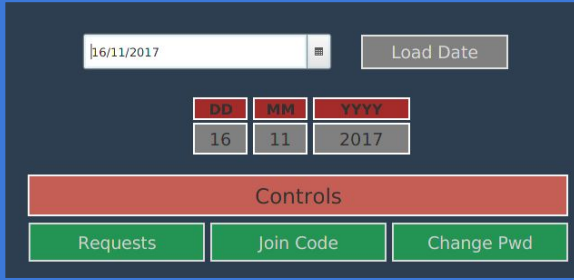
The screenshot shows the 'Student Controls' interface. At the top, there is a date input field with '16/11/2017' and a 'Load Date' button. Below this is a date picker with 'DD' (16), 'MM' (11), and 'YYYY' (2017) buttons. A red bar labeled 'Controls' is in the center. At the bottom, there are three green buttons: 'Time Table', 'Add Course', and 'Change Pwd'.

Student Controls



The screenshot shows the 'Faculty Controls' interface. It has the same top date input and 'Load Date' button. The date picker below shows 'DD' (16), 'MM' (11), and 'YYYY' (2017). The red 'Controls' bar is in the center. At the bottom, there are three buttons: a dark grey button on the left, a green 'Join Course' button in the middle, and a green 'Change Pwd' button on the right.

Faculty Controls



The screenshot shows the 'Admin Controls' interface. It features the same top date input and 'Load Date' button. The date picker below shows 'DD' (16), 'MM' (11), and 'YYYY' (2017). The red 'Controls' bar is in the center. At the bottom, there are three green buttons: 'Requests', 'Join Code', and 'Change Pwd'.

Admin Controls

Distribution of work

Nihesh Anderson

- **Helper Classes** - Room, Reservation, Course, SpamFilter, BookIT Constants.
- **GUI Classes** - Student, Faculty and Admin GUI and their controllers.
- Generated and parsed CSV timetable
- Implemented SpamFilter using Naive Bayesian Classifier model.
- Set-up server using socket programming.
- Creating setup files for resetting server database
- Implemented Automatic GUI scaling to suit screens with variable dimensions, on full screen mode.
- Generated executable JAR for one click launch on windows and linux platforms with Oracle Java 8.
- Tested the app, and fixed glitches/bugs.

Harsh Pathak

- **Helper Classes** - User, Faculty, Student, Admin, Email, Logged Out Exception Class.
- **GUI Class** - fxml + Controller
- Created css files for the Login Signup GUI
- Generated the app database for join codes, student requests, course postconditions, batch years, etc.
- Created fxml files for Login Signup to fit different screen specifications.
- Implemented joining code facility in admin class.
- Added Change password functionality for all users.
- Tested the app out for any glitches/ bugs.
- Made the javadoc for classes in the App.
- Made .exe executable of app for windows.