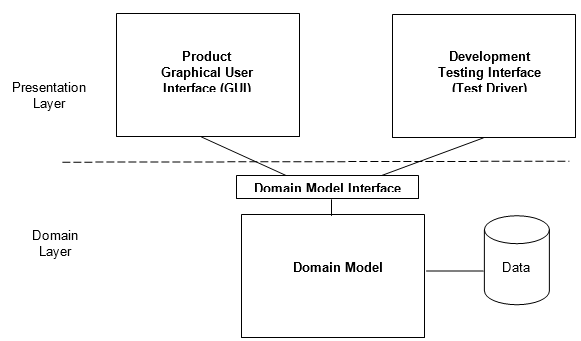
**Product Design**

|  |  |
| --- | --- |
|  | **Team C Thepiratebay.se** |

# Architectural Overview

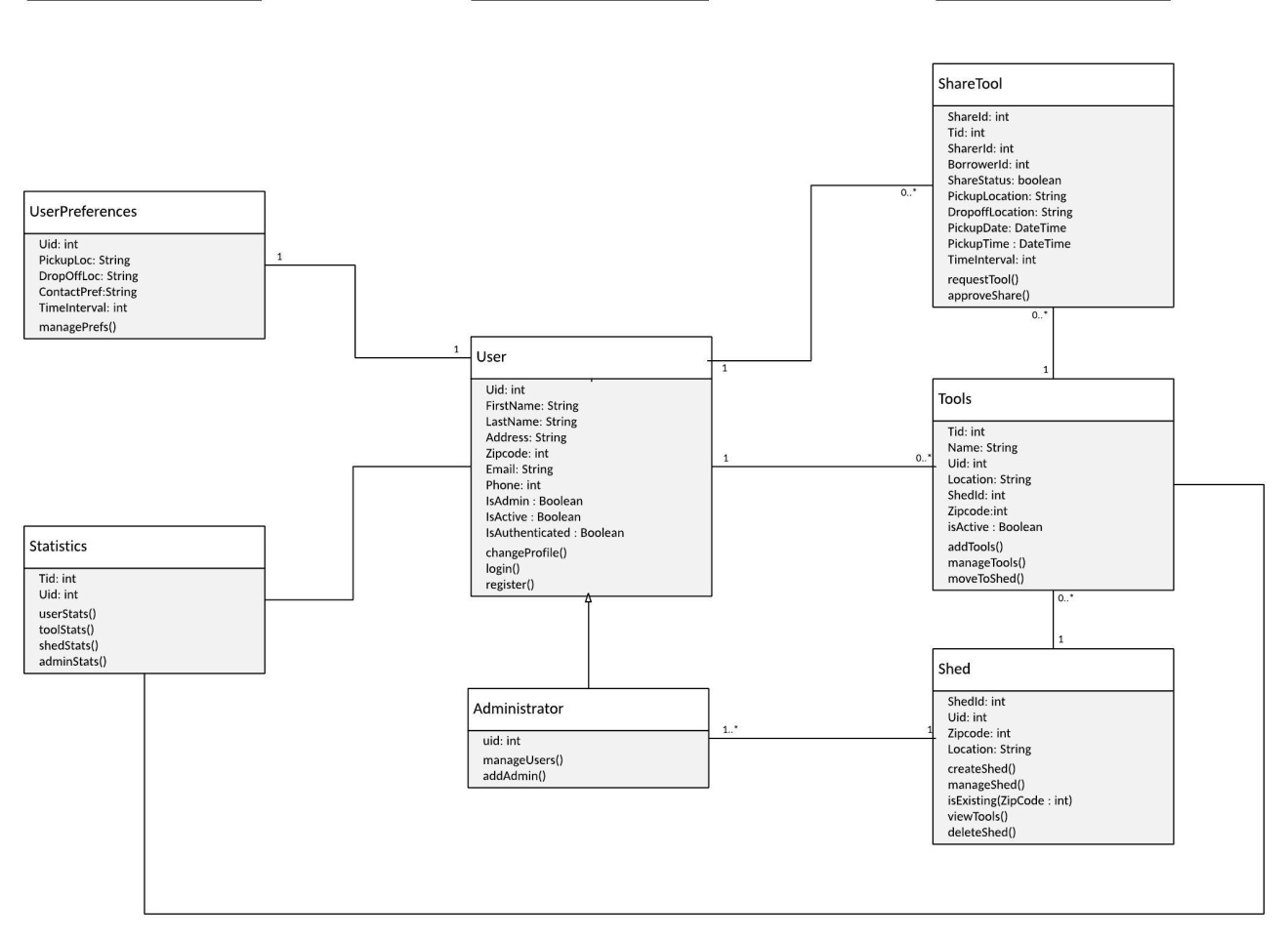


The Architecture is mainly composed of the presentation layer and the domain layer, the presentation layer is composed of the templates and forms module , while the domain model is composed of the model module. The logic of the system is controlled by the views module

# Components and Functions

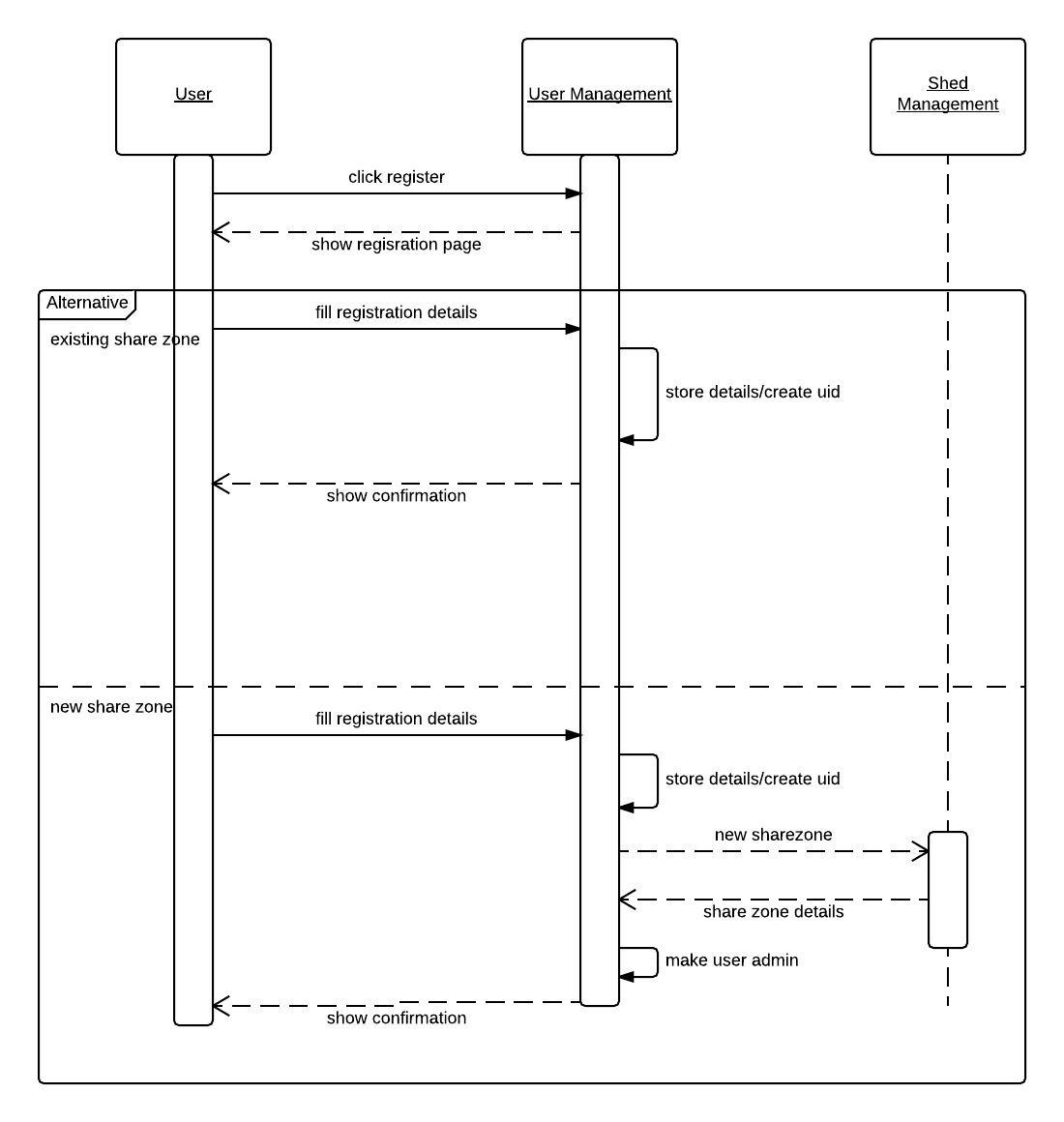
|  |  |
| --- | --- |
| Authenticate | States:   * Registration system requests for the user’s **email** which is used as a username, each user also needs a **Password.**   Behavior  The registration system performs the following functionalities:   * **Login()** for existing user * **Register()** for new user |
| Tool Management | States:   * The tool management system stores the following details related to the tools such as t**ool\_id, tool\_name, tool\_description**. * It is also responsible for maintaining the **rating** of the various tools present at that moment in the shed.   Behavior  The tool management system performs the functionalities of registering the tool, deactivating the tool, and blacking out the tool:   * **Register\_tool()** when a user want to allow other users to borrow the tool * **Deactivate\_tool()** when the user wants to remove the tool from the system * **Blackout\_tool()** when the user does not want to lend the tool for a specified set of days |
| Lending | States:   * The lending component registers a tool with the **tool\_id** * The **lender\_id** is also used to store the lender information * The t**ime\_limit** attribute is used to specify the duration for which the tool can be lent.   Behavior  This component enables the listing of the tool, moving of the tool to the shed and granting the tool:   * **list\_tool()** to list a tools shared by the user.. * **move\_toShed()** When the lender wants to move the tool to the shed. * **approve\_tool()** When the tool’s owner grants permission to a user to borrow the tool |
| Borrowing | States:  The borrowing component identifies a tool with the **tool\_id**  **Borrowing\_id** is used to identify the borrower  **Date** is used to specify the date on which the tool was borrowed  Behavior  The borrowing component enables the customer to request a tool and return the tool:   * **Request\_tool()** when a borrower requires a tool. * **Return\_tool()** when the borrower returns the tool after using it. * **Request\_Permission()** the borrower needs the permission of the owner prior to the tool being granted, this the done only if the tool is shared from a place other than the shed. |
| Shed\_Creation and Management | States:  This component stores information such as the **user id** which is used for identifying the registered administrator.  It also stores information such as the **shed location** and the **zip code**.  Behavior  It supports the functionality of creation of the shed:   * **Create\_Shed()** to create a community shed in a zipcode. there can only be a single shed in a zipcode. * **Add\_tool()** to add a tool to the shed. * **Remove\_tool()** to remove a tool from the shed. * **View\_tools()** the administrator can view the tools currently present in the shed. * **Black\_out\_days()** the administrator can specify days during which tools in the shed will not be lent. |
| UserManagement | States:  The user management component manages information related to the users such as **user id** and **rating** of the user.  Behavior  the administrator can perform the functions of banning the user from the system and adding new administrators.:   * **User\_banning()** the administrator can ban users in extreme cases. * **Add\_Admin()** the administrator can make other users as administrators. |
| Statistics | States:  The statistics component provides the **rating** of the user.  Behavior  View the user statistics by any user and administrator and view the the global statistics about all the user activities in the Zip Code by the administrator:   * **View\_globalstats()** by which an administrator can view statistics about the zipcode. * **View\_userstats()** by which a user can view their own statistics. |
| Edit\_profile | States:  The users can change their preferences like their **pickup\_location** , **drop\_off\_location,**  contact information like **email, address, phone**.  Behavior   * **Edit\_profile()** is used to update the user’s preferences. |

# Class Diagram(s)

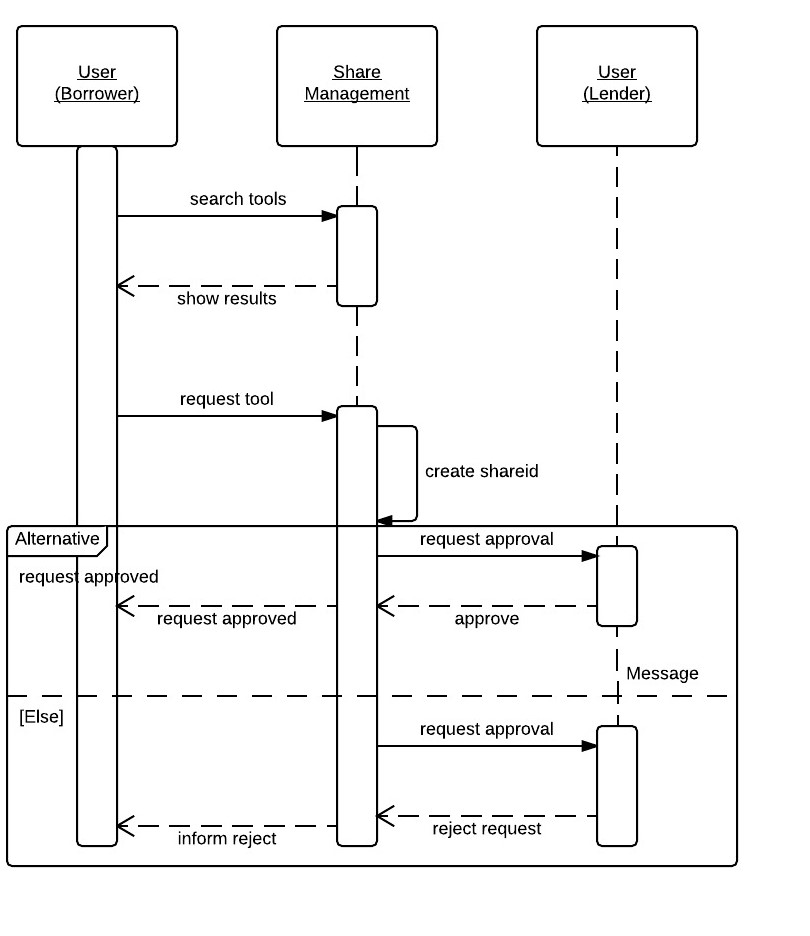


# Sequence Diagram(s)

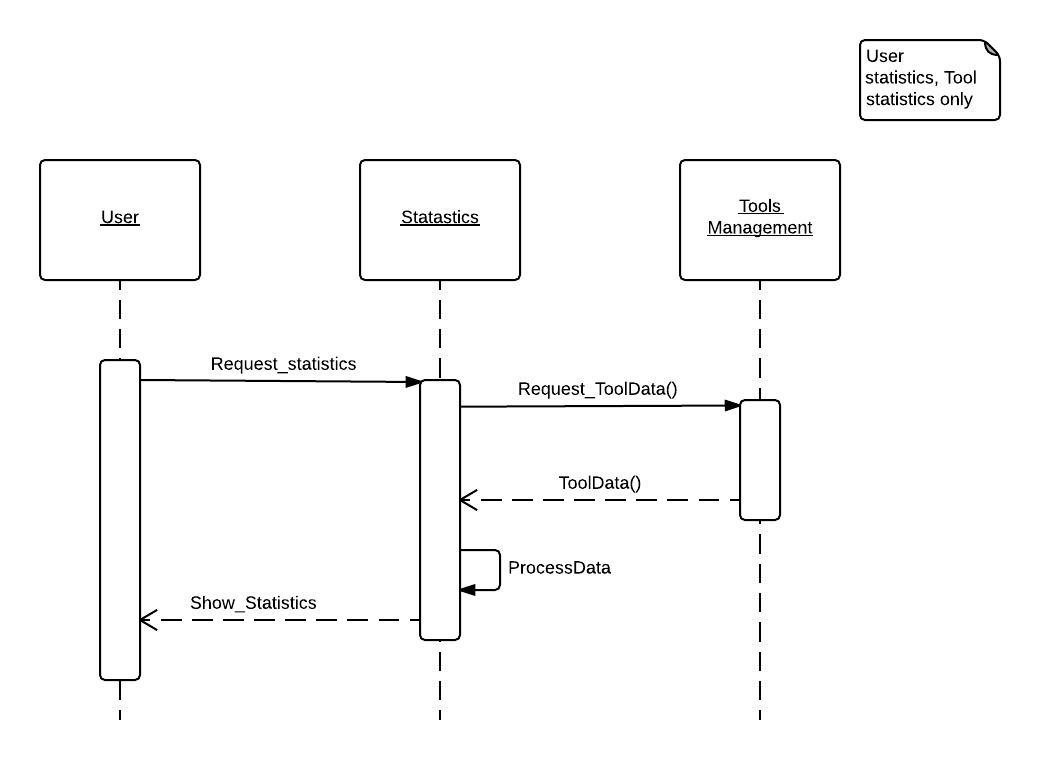
**1View Statistics (Admin)**



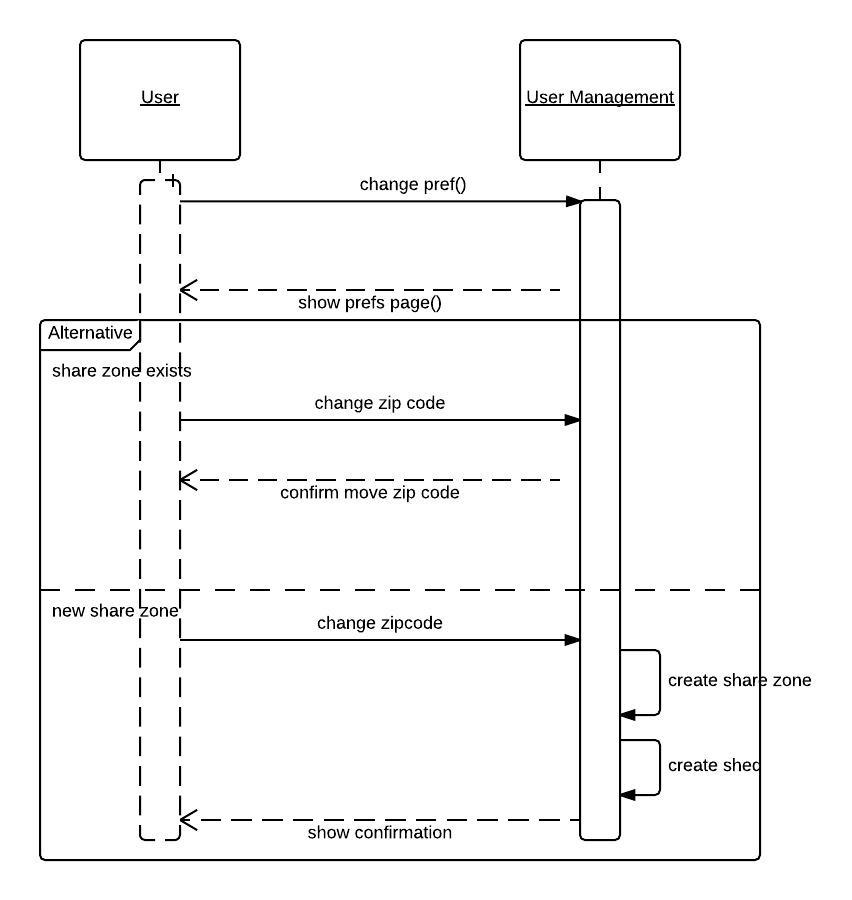
**2User Registration**



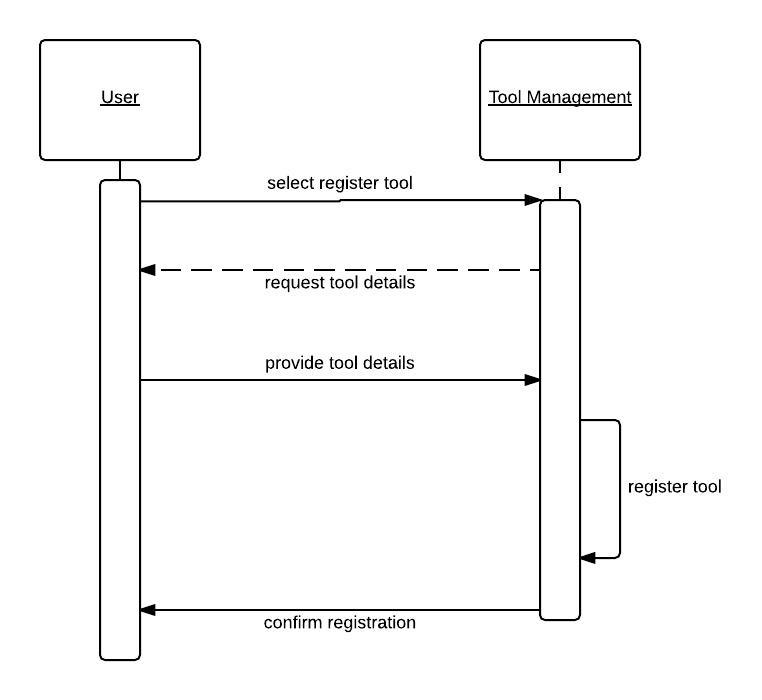
**3Creating a Tool Share**



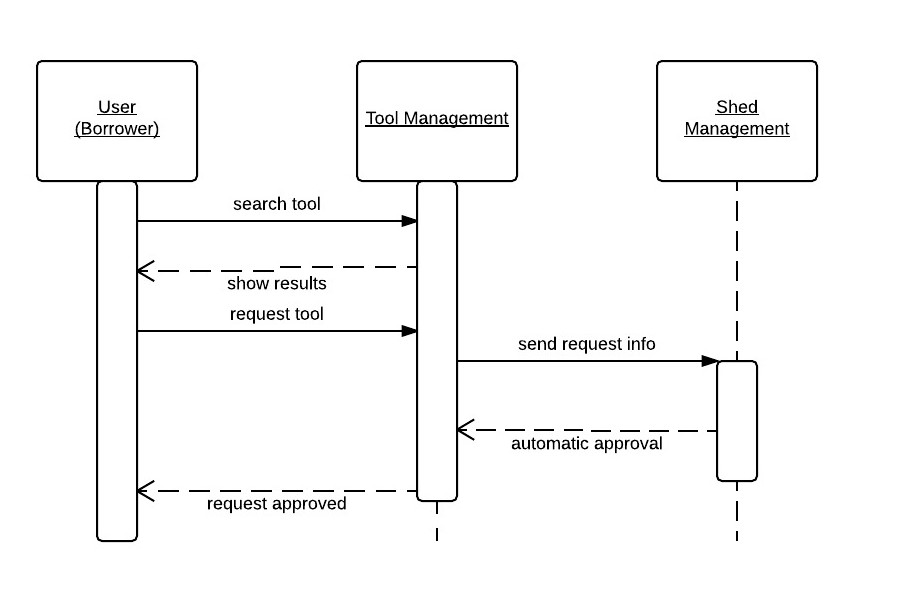
**4View User Statistics**



**5Change Preferences**



**6Tool Registration**



**7Sharing a tool from the shed**

# Design Rationale

For this project, we have decided to use the iterative approach. We are using the iterative approach as we can build a small itinerant of the system initially and build upon the system based on the customer feedback. One of the alternatives which was waterfall model was not suitable as the system might change as our understanding of the requirements changes.

**Update on 12-5-15** : We are now using the prototype model to build the project because we can build a working model very quickly and then develop a much better product which includes lots of additional features apart from the features advocated in the prototype. We have shifted to prototype model as it allows us-the development team a chance to get used to the development environment and also to learn the technologies being used to develop the web app.

We also decided to club together lending and borrowing operations into a single class called “share\_tool”, this was done as many of the attributes and methods are common, all the users of our system have the ability to borrow tools and the lend features can be added to users when they decide to list their tool in the system.

We decided to make all tools to have a unique id. All the tools also have the corresponding uid of the lender and borrower, this was done so that calculating statistics about the tool would become easier.

In our registration process, we also inform the user of the default arrangements for lending tools which can be changed later, this was done so that all the tools have a predefined location to be picked and dropped off at. The user can change the default arrangements later by accessing the user preferences option.

We have also added a security component so that users can only access their own accounts and profiles. Our admin can ban users depending on the users behavior. Admin can also make other users admin if he desires to. The admin is allowed to change the location of the shed, the email of the shed as well as the name of the shed. The admin of a share zone cannot leave the zone unless and until at least one another admin is available for that shed.

While showing the flow of sequence in the sequence diagram we have separated sequences of lending from the shed and lending from a user(who is sharing from his house) to make the diagrams simpler as keeping them in the same sequence will lead to multiple alternative flows which in turn makes the diagram look complicated.

Few of the methods such as addtools(), managetools() are kept in a separate class called as tools and are not kept in user class itself so that the user class is not too cramped. Another class for user preferences is maintained to change details of tool pickup and drop-off locations, this can be done in the user class itself but again this will make the user class more congested and convoluted.