**User Transaction System** ShedController ToolController name : String owner : ForeignKey borrower : ForeignKey register\_shed(request):response shed: ForeignKey view sheds(request):response description : string register\_tool(request):response update\_shed(UpdateView): response display\_tools(request):response is\_active : boolean community : ForeignKey borrowed\_tools(request):response **User Management System** tool(request, Tool\_id):response pickup\_arrangements: string deactivate\_tool(request, Tool\_id):response is\_requested : bool deactivate\_tool(request, Tool\_id):response requesting\_borrower : Foreignkey Shed update\_tool(UpdateView):response Session-Controller remove(self, res) : boolean remove\_tool\_from\_shed(request, tool):response return tool() : void name: string reserve(reservation, borrower) : void  $community\ \bar{\cdot}\ For eign Key$ create(user) : void address: string get\_address(user) : address admin: ForeignKey -shed adminis\_admin(user) : void is\_home : boolean login\_view(request) : response activate(self): void log\_out(request) : response ReservationController loggedin(func,\*args,\*\*kwargs) create(admin): void ourUser :: django.contrib.auth.models.user get\_tools: Tool[] 0..\* name : string reserve\_tool(request, tool):response 0..\* Reservation username : string return tool(request):response UserController address: string approve return(request, res id, mess id): response tool: ForeignKey zipcode: string display requests(request): response community : ForeignKey password : string accept request(request, res id): response borrow\_request\_date: DateTime community: ForeignKey register(request): response deny\_request(request, res\_id): response return\_date : DateTime Community is\_active : boolean update\_user(request): response retutned\_on : DateTime reputation : int home\_view(request): response borrower : ForeignKey zipcode: string community\_users(request): response owner : ForeignKey admin: ForeignKey my\_user(request): response is active : boolean name : string join(self, community): void show\_user(request, user\_id): response is\_complete :boolean is\_active : bool reputation(self): int feedBack(request, reservation\_id,mess\_id): message sent: boolean switch\_shed\_admin(self): void skipped : boolean chain mail(admin, user): void change\_password\_view(request): response feed\_back\_given : boolean add\_user(user) : void Statistics reason: string list available(): Tool[] 0..1 approved: boolean make community(user) : void community : ForeignKey admin\_change(user): void validate\_borrow(value) validate\_return(value) validate\_availability(value) most\_active\_reservation(): send\_deny(content): void Reservation send\_accept(self): void most\_active\_poster(): User FeedBack community\_change(self): void most\_recent\_res() : Reservation[] Admin :: User generate() : Object[] CommunityController user: ForeignKey rater: ForeignKey reputation: int chose\_community(request): response ban\_user(id) : void comment: string is\_comm\_admin(self): boolean select(request, community\_id): response is\_active: boolean update\_shed(UpdateView): response is\_shed\_admin(self): boolean new\_community(request): response remove\_user(id): void new\_admin(request): response update\_community(UpdateView): response post(self, reservation, mess\_id): void stats(request): response search\_objects(request): response Community Post System **Mailing System** CommunityPost Controller create\_community\_wall(request) : response MessageController Message display posts(request) : response display\_post(request, post\_id) : response Post add post(request) : response date\_sent : DateTime CommunityWall delete\_post(request, post\_id) : response subject: String post title : String contents: String timestamp : datetime send\_message(request):response wall name: string sender: ForeignKev poster\_id : ForeignKey view\_message(request message\_id): response community\_id : ForeignKey receiver: ForeignKey wall: ForeignKey content : String inbox(request) : response sent\_messages(request): response delete\_message(request, message\_id): response create(self, ourUser) :void post handler(self, t, user): int \_\_str\_\_(self) : String delete\_wall(self, t): void delete(self) : void edit(self, t, \*\*kwargs) : void